Pattern Discovery in Time Series

A survey

Denis Khryashchev

1Department of Computer Science
The Graduate Center of the City University of New York
dkhryashchev@gradcenter.cuny.edu
September 30, 2018

Abstract

Ordered sequences naturally appear in numerous data sets acquired from almost every major branch of science. They include medicine (physiological series, electrocardiography), natural language processing (sentences, texts), transportation engineering (traffic flow), weather forecasting (wind speed), economics (shares, futures, currency), and many others. Understanding the structure of time series can help detect abnormal heart rate, find similarities between sentences, forecast traffic flow, and stock market prices.

The main motivation for the survey is due to time series being ubiquitous and an integral part of most of the modern databases. Mining patterns in data sets with temporal order can help improve indexing, compression, and forecasting of the future events through better parameter selection, as well as determine similar records and partition the data.

Pattern discovery in time series relies on the techniques and methods used in machine learning, signal processing, and algorithmic information theory. They are applied to sequential data and include but not limited to analysis of complexity and predictability, search for recurring patterns and periodicity, similarity quantification and clustering, forecasting, and decomposition.

INTRODUCTION

A great many data sets capture continuous processes that go on for a significant period of time and produce some numeric output at regular time intervals. For example, the New York City’s bike-sharing system Citibike collects trip data for every bike recording the time and stations where the trips originated and ended. Aggregation of all the trip records that originated on the same station within consecutive 30-minute intervals leads to time series of bike demand. Such time series is a sequence of non-negative integer values ordered in time.
Periodicity and seasonality are among the most important characteristics of time series that could help better understand the underlying process that generated it and improve forecasting accuracy. For example, given the turnstile data of New York City subway we can identify that the weekday morning rush hours happen daily between 6:30 and 9:30 AM, and look for other periodic patterns. Figure 1 shows annual average weekday trends for subway, vehicles for hire, and taxi ridership in New York City. As a first-order of approximation we can expect a future number of passengers at 7:00 AM to be similar to the average of the previously observed passenger counts at the same time in the past. Chen et al. [1] identified 5 strong diurnal patterns occurring in subway transit system of New York City.

Another important problem that arises in time series analysis and forecasting is quantification of time series complexity and predictability. For instance, given time series of a healthy individual’s electroencephalogram (EEG) we can quantify its complexity as a baseline, i.e. assign a numeric value to its internal structure. In the future monitoring new EEG we will be able to detect irregularities and abnormalities if their complexity is significantly different from the baseline. Petrosian [3] demonstrated that it is possible with the approximation of Kolmogorov complexity. Figure 2 shows 5-second samples of EEG time series that represent interictal, preictal, and seizure stages.

Predictability, being a derivative of complexity, can help us estimate the expected theoretical accuracy of a forecast given the series. It could be very useful in case of a trade-off between the accuracy and running time of a certain forecasting algorithm. Zhao et al. [4] ordered time series in terms of their theoretical maximum predictability and showed that the accuracy of various forecasting algorithms improves with the increase of the value of predictability with the latter being a theoretical limit of accuracy.

Building stock market portfolios traders and investors face a problem of quantification of time series similarity or time series clustering. To illustrate, we assume that a trader knows the behaviour of a certain stock market
index and would like to select shares that are not included in the index calculation but exhibit a similar behavior. E.g. their value grows and decreases simultaneously or at equivalent periodic intervals. Fu et al. [5] applied self-organizing map clustering to determine similarities among Hong Kong’s stock market index Hang Seng and a few other stock time series.

Most of the mentioned techniques and methods of time series data mining are quite often used to improve forecasting. For example, Sun et. al [6] decomposed the data that represent subway passenger flow into a collection of time series with different frequencies with wavelet transform and then trained a support vector machine model for every series. Another work that incorporates time series mining into forecasting was done by Espinoza et. al [7]. In it electrical substation load series were first analyzed for seasonality and periodicity. Due to the presence of strong daily, weekly, and monthly periodic patterns, they were able to select proper parameters for autoregressive models and to forecast the future values of the substation load.

In this survey time series data mining techniques including analysis of complexity, entropy, predictability, seasonality and periodicity, similarity quantification and clustering are reviewed. Forecasting, various measures of auto-correlation are also presented. Figure 3 shows the overall organization of the survey and relationship between the chapters. Measures of periodicity, predictability and entropy are considered as approximations of complexity, and as methods that help in model and parameter selection for time series forecasting. Measures of similarity or distance are used for clustering and forecasting.

The concepts are grouped together in the following order: chapter 1 describes complexity, entropy, and predictability; chapter 2 covers periodicity; chapter 3 is on measures of dependency and clustering; chapter 4 finalizes the survey with the outline for the most widely used time series forecasting techniques.
We will only consider mining of univariate time series data with real values denoted with braces \{\} (not to be confused with set notation). I.e. an ordered sequence $X = \{X_1, \ldots, X_N\} = \{X_t\}_{t=1}^N, X_t \in \mathbb{R}$ that represents a series of $N$ measurements of the same variable collected over time. We will denote subseries that start and end at time intervals $i$ and $j$ as $X^j_i = \{X_t\}_{t=i}^j$. We will also study a collection of such series to measure pairwise similarity and perform clustering.

I. Complexity, Entropy, and Predictability

In the first chapter we overview the ways to quantify complexity of a time series. Complexity can provide us with the better understanding whether a given time series is easier to forecast than others, measure the amount of information in it, and estimate the theoretical accuracy of a forecast, i.e. it can be approximated in terms of entropy and predictability.

A. Definition of complexity

Estimation of complexity of a time series has been the focus of a considerable attention. Naturally, complexity of a time series identifies the bounds of its theoretical predictability, i.e. limits the maximal possible average accuracy of a forecast one could achieve with a perfect predicting algorithm or technique. In other words, the more complex or more chaotic a time series is the harder it is to predict its future values.

As an illustration of the complexity concept, consider two simple binary time series $X$ and $Y$ illustrated on Figure 4:

\begin{align*}
X &= \{0, 1, 0, 1, 0, 1, 0, 1, 0, 1\} \\
Y &= \{1, 0, 0, 1, 1, 1, 0, 0, 1, 0\}
\end{align*}
In these series both 1 and 0 appear equally likely

\[ P(X = 1) = P(X = 0) = P(Y = 1) = P(Y = 0) = 0.5 \]

Both series have the same expected value \( E(X) = E(Y) = 0.5 \) and variance \( \text{Var}(X) = \text{Var}(Y) = 0.25 \), however, we can clearly notice that \( X \) has a very basic periodic behavior while \( Y \) seems to be more chaotic and complex. Kolmogorov was among the first to propose a universal theoretical measure of complexity [8] that could be applied to sequences.

For time series we define the Kolmogorov complexity as the length of the shortest description of a sequence of values ordered in time in some fixed universal description language

\[ K(X) = |d(X)| \]

where \( K \) is the Kolmogorov complexity, and \( d \) is the shortest description of the time series \( X \). For example, if we wanted to estimate complexity of the sequences depicted on Fig. 4 \( X \) and \( Y \), and selected Python as the description language \( d_P \), then we could end up with two corresponding descriptions \( d_P(X) = [0, 1] * 5 \) and \( d_P(Y) = [1, 0, 0, 1, 1, 0, 1, 0] \). First description creates a list of 5 repeating subsequences \( \{0, 1\} \) that match \( X \) while the second description simply lists all the elements of \( Y \). Having the length of the first one equal to 7 and the second one being equal to 21 (counting every symbol in the descriptions), we can conclude that \( K(X) = 7 \) and \( K(Y) = 21 \) in terms of \( d_P \). Clearly, time series \( Y \) is more chaotic and complex.

However, the key limitation of the Kolmogorov complexity in practice is it has been proven that one cannot compute the true Kolmogorov complexity [9] [10] in general and instead it is usually approximated with entropy or some similar computable analogue.

Kolmogorov complexity can be approximated with Kolmogorov entropy [11]

\[ K = - \lim_{\tau \to 0} \lim_{\epsilon \to 0} \lim_{d \to \infty} \frac{1}{d\tau} \sum_{i_1, \ldots, i_d} p(i_1, \ldots, i_d) \ln p(i_1, \ldots, i_d) \]
which describes a complexity of a dynamic system with $F$ degrees of freedom. The $F$-dimensional phase
space with an attractor is partitioned into $\epsilon^F$ boxes, and the time series $X$ being a trajectory in the space is in
the basin of attraction. $\tau$ measures the time intervals and $p(i_1, \ldots, i_d)$ is the joint-probability that $X_{t=k\tau}$ is in the
box $i_k$. The main disadvantage of the definition and its analogues is it provides a straightforward approach to
compute complexity for a well-defined analytical model and does not allow for computation of the complexity
when the model is unknown.

One of the most well-known approximation of Kolmogorov complexity for a finite sequence or time se-
ries $X$ was proposed by Lempel and Ziv [12] and it is based on the notion of production history. Production
histories are sequences of subsequences of $X$ that can be used to reconstruct it. For example, for se-
ries $X = \{0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1\}$ one of the production histories would contain subsequences
$\{0\}, \{0, 0\}, \{1, 0\}, \{1, 0, 0\}, \{1, 0, 0, 0\}, \{1, 0, 1\}$. They defined complexity of a sequence as the length of its
shortest production history

$$c(X) = \min \{c_H(X)\}$$

where $c_H$ stands for the number of components in a history of production of the sequence $X$, i.e. a sequence
is considered more complex if it required more steps to produce it. However, the disadvantage of this definition
is that numbers are treated as symbols and for time series $X = \{1, 2, 1, 2, 2, 1\}$ and $Y = \{1, 15, 1, 15, 15, 1\}$ we have $c(X) = c(Y)$ which might not be suitable for certain predicting algorithms.

A similar definition of complexity approximation was proposed in the work of Charikar et al. [13] where it is
estimated for a string with matching the smallest context free grammar that reconstructs it.

B. Applications of complexity

Batista et. al [14] proposed new complexity-invariant distance measure for time series. They noticed that in
certain cases distance calculated for time series of high complexity that represent similar objects like leaf or skull
shapes and are very similar appear to be further apart when compared to dissimilar series of lower complexity.
They demonstrated that existing distance measures that are invariant in translation, amplitude and offset, local
scaling or “warping”, uniform scaling, phase, and occlusion are not efficient to measure distance between complex
time series. As an alternative a complexity-invariant distance measure (CID) between two time series $X$ and $Y$
was proposed

$$CID(X, Y) = \|X - Y\| \max(CE(X), CE(Y)) / \min(CE(X), CE(Y))$$

where $CE$ is the proposed complexity estimator

$$CE(X) = \|X^N - X^{N-1}\|.$$

Alternative application of complexity was introduced by Hyvärinen [16] who proposed an alternative to standard Principal Component Analysis and Independent Component Analysis (PCA [17] and ICA [18]) for time series. Both of the analyses do not take the temporal order in a time series into account. To take the order into account, the proposed approach looks for such projections $Y$ of the original time series $X$ that the approximation of the Kolmogorov complexity of the projections $\hat{K}(Y) = \sum_t H(Y - \hat{Y})$ is minimal. $H$ denotes the measure of entropy, $\hat{Y}$ is a result of forecasting $Y$ with some forecasting algorithm $f$, $\hat{Y}_t = f(Y_{t-1}, Y_{t-2}, \ldots, Y_1)$. The disadvantage of the approach is the trade-off between the accuracy of the estimation of the complexity of the projection $Y$ and complexity of the predictive algorithm: the more complex the algorithm, the harder it is to find the optimal projection.

Other direct and indirect approaches to approximation of the complexity of a time series are estimation of entropy, predictability and periodicity. Roughly speaking, a time series with lower entropy, higher periodicity, and higher predictability is less complex.

C. Entropy

Entropy is an effective measure that quantifies the amount of regularity and internal structure in a time series. It allows to estimate the amount of information such series carry about its underlying generating process. Entropy increases with the degree of disorder of time series data, it is at its maximum for completely random and unpredictable systems. A high entropy usually means low predictability and high complexity.

Entropy was originally introduced in information theory by Shannon and Weaver [19] who aimed to use it as a measure of information transmitted by a signal in some communication channel. They defined it as

$$H(X) = \mathbb{E}[-\log_2 P(X)] = -\sum_t P(X_t) \log_2 P(X_t)$$

(9)

Rényi generalized Shannon’s entropy [20] in the form $H(X) = H(\mathcal{P})$ (with $\mathcal{P} = (p_1, \ldots, p_N)$ being a generalized probability distribution of $X$) modifying 4 postulates proposed by Fadeev into 5. Assuming our time series $X$ has ordinary discrete finite distribution $\mathcal{P}$, i.e. $\sum_i p_i = 1$ Renyi entropy of order $\alpha$ is

$$H_\alpha(X) = H_\alpha(\mathcal{P}) = \frac{1}{1 - \alpha} \log_2 \left( \sum_k p_k^\alpha \right)$$

(10)

where for $\alpha \to 1$ it is equal to Shannon entropy $H(X)$. Both the Shannon definition of entropy and its generalization by Rényi are not able to capture temporal patterns and order in time series and would measure
the same amount of information for a time series and its randomly scrambled copy. I.e. $H_α(p_1, p_2) = H_α(p_2, p_1)$ due to its symmetry. A time-correlated entropy was defined to address this problem and to take not only the frequencies of specific values, but also the order their appearance in the series. It is defined as $H_t(X) = -\sum_{i=1}^{N} \sum_{j=i}^{N} P(X^i_j) \log_2 P(X^i_j)$ (11)

$P(X^i_j)$ represents the probability of finding a particular time ordered subsequence $X^i_j$ of the time series $X$ that starts at time interval $i$ and ends at time interval $j$. Unlike Shannon entropy, the time-correlated entropy considers not only the frequency of the appearance of the values $X_t$, but also their order capturing the temporal patterns of the time series $X$.

The problem of finding all the subsets of a given set has exponential complexity $O(2^n)$ creating computational difficulty of calculating entropy for long time series. One way to reduce the complexity is to use the Lempel-Ziv estimator. It was shown that it can rapidly converge to the real time-correlated entropy $H_t(X) \approx H_{LZ}(X)$. Lempel-Ziv estimator of the time-correlated entropy is defined as

$$H_{LZ}(X) = \left( \frac{1}{N} \sum_t s'_t \right)^{-1} \ln N$$ (12)

where $s'_t$ represents the length of the shortest subsequence starting at the time $t$ which does not appear from the beginning of $X$ to $X_{t-1}$. The definition is limited for the approximation of entropy of short time series: the shorter the series the less accurate the entropy estimation.

Alternative way to define time-correlated entropy that is not limited in accuracy for smaller time series and able to capture temporal patterns was proposed by Bandt and Pompe [23]. They introduced the permutation entropy of order $n$

$$H(n) = -\sum p(\pi) \log p(\pi)$$ (13)

where $p(\pi) = \frac{\# \{0 \leq t \leq T \text{type}(x_{t+n+1}, \ldots, x_{t+1}) = \pi \}}{T-n+1}$ is the relative frequency of a permutation $\pi$ of order $n$. The permutation $\pi$ stands for a well-ordered subsequence of $X_t$ of length $n$. The permutation entropy estimates the information that contains in the relative frequencies of all permutations of order $n$.

For example, if we consider time series $X = \{4, 7, 9, 10, 6, 11, 3\}$ and $n = 3$, then We have the permutations $\pi(4, 7, 9) = \pi(7, 9, 10) = \pi_{012}(X_t < X_{t+1} < X_{t+2}), \pi(9, 10, 6) = \pi(6, 11, 3) = \pi_{201}(X_{t+2} < X_t < X_{t+1})$, and $\pi(10, 6, 11) = \pi_{102}(X_{t+1} < X_t < X_{t+2})$. The resulting entropy becomes $H(3) = -2 \frac{2}{5} \log \frac{2}{5} - \frac{1}{5} \log \frac{1}{5} \approx 1.5$.

Two major limitations of the proposed permutation entropy are the requirement for all the neighboring values to be different, $X_t \neq X_{t+1}$, and the explosive growth of the total number of permutations ($n!$) to test. Fadlallah [24] proposed a generalization that takes into account the amplitudes of the time series $X$ or actual values of $X_t$ through introduction of weights for every extracted subsequence that matches a certain pattern.
Yet another measure of complexity designed to take the temporal order of time series into account and to have a low computational complexity is the approximate entropy proposed by Pincus [25]. To estimate it, we fix two parameters $m$ and $r$ and calculate the approximate entropy as

$$ApEn(m, r) = \lim_{n \to \infty} \left( \Phi^m(r) - \Phi^{m+1}(r) \right)$$

(14)

where $\Phi^m(r) = \sum_{i=1}^{N-m+1} \frac{\log C_i^m(r)}{N-m+1}$, the correlation $C_i^m(r) = \frac{|\{j \in N-m+1 : d[X^{(i)}, X^{(j)}] \leq r\}|}{N-m+1}$ with the distance measure $d[X^{(i)}, X^{(j)}] = \max_{k=1,\ldots,m} (X_{i+k-1} - X_{j+k-1})$, and $X^{(i)} = \{X_i, \ldots, X_{i+m-1}\}$. It is not clear how to pick the parameters $m$ and $r$ and such a definition of entropy requires additional analysis of the data to pick the parameters.

D. Predictability

Maximum Predictability is a natural derivative measure of complexity based on the amount of entropy. Following the Fano inequality [26] we have

$$\Pi(X) \leq \Pi^{\text{max}}(H[X], N)$$

(15)

i.e. predictability $\Pi(X)$ of the series $X$ is limited by $\Pi^{\text{max}}$ given the selected measure of entropy $H[X]$ and the number $N$ of unique values $X_t$. Adapting the work done by Song et al. [27] and selecting Lempel-Ziv approximation of the time-correlated entropy $H_{LZ}(X)$, we have Maximum predictability $\Pi^{\text{max}}$ being equal to the solution to the equation

$$H_{LZ}(X) = -\Pi^{\text{max}} \log_2(\Pi^{\text{max}}) - (1 - \Pi^{\text{max}}) \log_2(1 - \Pi^{\text{max}}) + (1 - \Pi^{\text{max}}) \log_2(N - 1)$$

(16)

Maximum predictability sets upper bound for the accuracy that can be achieved with an arbitrary predicting algorithm $\alpha$ to forecast the value $X_{t+1}$ given $\{X_1, X_2, \ldots, X_{t-1}\}$. The value $\Pi^{\text{max}}$ takes values on $[0, 1]$ where 0 stands for complete unpredictability and randomness, while 1 stands for 100% predictability. For example, for time series that consists of only one repeating value $X = \{1, 1, 1, 1, 1, \ldots, 1\}$ we have time-correlated entropy $H_{LZ}(X) = 0$ and maximum predictability $\Pi^{\text{max}} = 1$. For more chaotic series the entropy grows decreasing the maximum predictability.

Solving the equation with numerical methods one can set the lower bounds on the expected errors in the series forecasting. Figure 5 demonstrates several distinct cases for different values of time correlated entropy $X_{LZ}(X)$ and number of unique values in the series $N$ and the resulting $\Pi^{\text{max}}$. With $f(\Pi^{\text{max}})$ we denote $-\Pi^{\text{max}} \log_2(\Pi^{\text{max}}) - (1 - \Pi^{\text{max}}) \log_2(1 - \Pi^{\text{max}}) + (1 - \Pi^{\text{max}}) \log_2(N - 1) - H_{LZ}(X)$. 

9
Accurate estimation of predictability of time series $\Pi^{\text{max}}$ strongly depends on the selected measure of the entropy. Among the most suitable for time series definitions are the Lempel-Ziv approximation of time-correlated entropy $H_{LZ}(X)$ and permutation entropy $H(n)$ proposed by Bandt and Pompe. Lempel-Ziv approximation quite often appears in papers related to forecasting of human mobility and transportation, e.g. Huang et al. [28], Song et al. [29]. The disadvantage of the measure of predictability $\Pi^{\text{max}}$ is due to its accuracy being defined by the selected measure of entropy $H(X)$.

Kaboudan proposed alternative approach to time series predictability based on generic programming [30]. The key idea is to compare errors of forecasting the time series before and after shuffling. The more the shuffling decreases forecasting accuracy, the more predictable is the series. The measure is defined as

$$\eta(X) = 1 - \frac{(X - f(X))^T (X - f(X))}{(X^s - f(X^s))^T (X^s - f(X^s))}$$  \hspace{1cm} (17)

where $X^s$ denotes shuffled time series $X$, $f(X)$ and $f(X^s)$ stand for predictions of values of $X$ made with a predictor $f$. One of the disadvantages of the $\eta$ measure is that its range shrinks for longer time series making it an inefficient measure. However, the main weakness of the approach is due to the fact that the shuffling of the series is done only once which could lead to different and potentially inconsistent results for the same time series $X$ if $\eta(X)$ is evaluated at different times. A better approach would be to shuffle the series numerous times (e.g. 1000) and calculate a $p$ value and performing hypothesis testing.

Yet another alternative approach widely used in econometrics and economics is based on the predictive

Figure 5. Some solutions for $\Pi^{\text{max}}$ given the time correlated entropy $S$ and number of unique values $N^{(i)}$
regression \cite{31}. The idea is to fit a regression model, e.g.

\[ X_{t+1} = \alpha + \beta X_t + \epsilon_{t+1} \quad (18) \]

and calculate \( R^2 \)

\[ R^2 = 1 - \frac{\text{Var}(\alpha + \beta X_t)}{\text{Var}(X_{t+1})} \quad (19) \]

\( R^2 \) quantifies the ratio of the variance that was explained with the model \( \text{Var}(\alpha + \beta X_t) \) and the total variance \( \text{Var}(X_{t+1}) \). \( R^2 > 0 \) signifies that \( X_{t+1} \) can be forecast given values of \( X_t \). And the bigger the value of \( R^2 \) is, the bigger is the overall predictability of the series \( X \). Having the advantage of simplicity, the approach, however, is not designed to capture non-linear patterns in time series.

**E. Summary**

Measures of time series complexity have a wide range of application, starting from classification \cite{3}, to data compression \cite{32}, complexity-invariant distance measures \cite{14}, and measures of predictability \cite{27}.

The universal measure of complexity proposed by Kolmogorov is not a computable function and it is usually approximated with entropy. In case of time series entropy measures that capture temporal patterns are based on counting permutations and evaluation of frequencies of appearances of subsequences \cite{21} \cite{22} \cite{23}. The main disadvantage of combinatorial measures of entropy are due to their exponential \( O(2^N) \) or factorial complexity \( O(n!) \), as well as their inability to capture the actual amplitudes. Some generalization of permutation entropy \cite{24} capture actual amplitudes but are still computationally expensive. Measures of predictability are either dependant on selected measures of entropy or on the regression techniques.

Therefore, future directions might include generalization of combinatorial measures of entropy similar to \cite{24} so they capture the actual amplitudes, and improvement of their computational complexity for they are impractical for very long time series.

**II. PERIODICITY AND SEASONALITY**

**A. Estimating periodicity**

It is a general understanding that time series that represent activity of a large group of people, e.g. electricity demand, water consumption, public transportation, etc. contain very distinct periodic patterns. For example, there are morning and evening rush hours in the subway of New York City, seasonal changes in electricity consumption (air conditioners in the summer).

Detecting periodicity in time series can improve parameter selection for the model used for its forecasting and speed-up the training process. Hoffman et al. \cite{33} proposed a simple lagged average predictor. If a series
If $X$ strongly correlates with itself at lag $\tau$, then the prediction through lagged average can be made as
\[
\hat{X}_{t+1} = \frac{\tau}{t} \sum_{i=1}^{\tau/t} X_{t-i\tau}.
\]

Moreover, periodicity of a sequence is directly related to its predictability. Fon-Der-Flaass and Frid [34] demonstrated that infinite sequences with high periodicity have low complexity. On the other hand, Lorenz [35] showed that non-periodic sequences are unpredictable.

The standard way to determine periods is to bring the values of time series $X$ into the frequency domain with a discrete forward Fourier transform [36]:
\[
M_k = \sum_{t=0}^{N-1} X_t e^{-i2\pi kt/N}
\]  
(20)

where $M_k$ represents the magnitude of the $k^{th}$ period quantifying “relative chance” of a repetition of the values of original time series. Due to the symmetry of the transformation: $M_{n+k} = M_k$ and $M_{k+n} = M_k$ it is possible to perform a fast Fourier Transform [37] obtaining
\[
M_k = \sum_{n=0}^{N/2-1} X_{2n} e^{-i4\pi kn/N} + e^{-i2\pi k/N} \sum_{n=0}^{N/2-1} X_{2n+1} e^{-i4\pi kn/N}
\]  
(21)

Analysis of the magnitudes $M_k$ provides an insight into whether there are periodic signals in the time series. Figure 6 depicts the results of Fourier transform of Yellow Taxi pick-up time series collected in midtown of Manhattan in July of 2014. We can clearly observe largest magnitudes for the periods of 8 and 12 hours, as well as 1 day and about 1 week.

The key disadvantage of periodicity estimation with Fourier Transform and its derivatives is that it’s not linear in period and adds spurious artifacts due to considering the values of the time series $X$ to be looped, i.e. the last value $X_N$ is followed by $X_1, X_2, \ldots$. 

![Figure 6. Frequency domain of Yellow Taxi time series](image)
To decrease the presence of spurious artifacts, various window functions are applied to Fourier Transform \cite{38} dividing the time series $X$ into smaller series localizing the transformation. Short-time Fourier transform is defined as

$$M_k = \sum_{t=0}^{N-1} X_t w(t - \tau) e^{-i2\pi kt/N} \quad (22)$$

where $w$ is the window function and $2\tau$ is equal to the effective width of the window. Essentially, Short-time Fourier transform passes portions of time series through a window into a regular Fourier transform. Numerous window functions have been proposed by Blackman, Hamming \cite{39}, Bartlett \cite{40}, and many others. A detailed survey of the trade-offs and proper window function selection was published by Harris \cite{41}. The trade-off between different window functions is between main lobe width and peak side lobe level of the signal, rate of the decrease of the side lobes, etc.

Fourier Transform is related to autocorrelation function through the theorem first proposed by Wiener \cite{42} and later generalized by Cohen \cite{43}. Given the time series $X$, $X_t \in \mathbb{R}$ the theorem states

$$S_{xx}(f) = \sum_{t=0}^{N-1} r_{xx} e^{-i2\pi kt/N} \quad (23)$$

where $S_{xx}$ stands for the power spectrum of $X$, $r_{xx}$ is its autocorrelation function.

Exploiting the relationship between autocorrelation and power spectrum one can estimate periodicity of time series $X$ with autocorrelation function \cite{44}. If a time series $X$ is exactly $p$-periodic, then its autocorrelation function $r_{xx}(\tau) = r_{xx}(\tau + p)$. In other words, values of the autocorrelation at time lags $\tau$ can be used as a medium for periodicity of the time series $X$.

A similar approach to test for the periodic signals in time series was proposed by Fisher \cite{45}. It is based on the notion of periodograms introduced by Schuster \cite{46}. A periodogram of time series $X$ used for Fisher’s test is computed in the trigonometric form:

$$I(f) = \frac{2}{N} \sum_{t=0}^{N-1} X_t \cos 2\pi ft + \frac{2}{N} \sum_{t=0}^{N-1} X_t \sin 2\pi ft \quad (24)$$

where frequency $f$ is limited by $-\frac{1}{2} \leq f \leq \frac{1}{2}$. Time series $X$ are viewed as a composition $X = \zeta + a$. $\zeta$ represents unobserved real signal that contains periodicity and $a$ is a normally distributed noise. The test evaluates the null hypothesis $H_0 : \zeta = 0$ meaning that there is no periodicity in the series which consists of noise only, $X = a$. The test is based on the statistic

$$W = \max_f I(f) \quad (25)$$

Fisher’s test for periodicity has caveats similar to those of Fourier transform and could be improved with the use of window functions.
An alternative approach to periodicity analysis of a time series was proposed by Sethares and Staley [47]. It allows for a linear-in-period transformation of a time series. They define time series \( X \) to be \( p \)-periodic if \( X_t = X_{t+p} \), and \( P_p \) to be a set of all \( p \)-periodic sequences. The key idea of the approach is to find a non-orthogonal set of \( p \)-periodic basis vectors and project the series onto them. They define \( p \)-periodic basis sequences as

\[
\delta^s_p(j) = \begin{cases} 
1 & \text{if } (j - s) \mod p = 0, \\
0 & \text{otherwise},
\end{cases}
\]

(26)

where \( s \) is the time shift and \( j \) is the time index of the series being analyzed. Then the measure of periodicity is the projection

\[
\pi(X, P_p) = \sum_{s=0}^{p-1} \alpha_s \delta^s_p, \quad \alpha_s = \frac{1}{N} \sum_{n=0}^{N-1} X_{s+np}.
\]

Figure 7 shows sample non-orthogonal basis sequences for \( p = 4 \).

<table>
<thead>
<tr>
<th>( j )</th>
<th>( \cdots )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \delta^0_4(j) )</td>
<td>( \cdots )</td>
</tr>
<tr>
<td>( \delta^1_4(j) )</td>
<td>( \cdots )</td>
</tr>
<tr>
<td>( \delta^2_4(j) )</td>
<td>( \cdots )</td>
</tr>
<tr>
<td>( \delta^3_4(j) )</td>
<td>( \cdots )</td>
</tr>
</tbody>
</table>

Figure 7. 4-periodic basis vectors. Source: [47]

Being able to calculate linear-in-period transformation of \( X \) that has a low computational complexity makes Periodicity Transform a good alternative to standard periodicity tests. On the other hand, the Periodicity Transform is inaccurate when the sample interval of the series \( X \) does not correspond to an integer factor of \( p \). Long periods are decomposed inexactly.

**B. Summary**

Majority of the methods that estimate periodicity of a time series are related to long- and short-time Fourier transform [37][38][41] or periodograms [45]. Their generalizations include Wavelet transform [48] that are based on construction of optimal piece-wise constant wavelets. However, the major limitation of this family of techniques is due to their non-linearity in period. Alternative approach based on projection of time series onto \( p \)-periodic basis sequences [47] is not accurate for larger periods and requires the sample interval of time series to be an integer factor of the period. Detection periodicity with linear autocorrelation function [42] does not detect non-linear periodic patterns in time series.

Therefore, potential research directions include the study of non-linear autocorrelation functions.
III. SIMILARITY AND CLUSTERING

In this section we will briefly describe the measures of similarity and dependence for time series including linear correlation coefficient and its generalizations with maximal and monotone correlation. In the second part of the chapter several major types of clustering techniques are discussed in their application to time series including methods that cluster raw data, generate features, and fit a model.

A. Correlation, measures of dependence

The most well known and widely used measure of dependence is linear or Pearson correlation coefficient. For two time series \(X\) and \(Y\) it is defined as

\[
\rho_{X,Y} = \rho(X, Y) = \frac{\mathbb{E}[(X - \mathbb{E}(X))(Y - \mathbb{E}(Y))]}{\sigma_X \sigma_Y} \tag{27}
\]

and is limited in the range \(-1 \leq \rho(X, Y) \leq 1\) with 0 standing for no linear dependence, -1 and 1 representing the strongest negative and positive correlation correspondingly. In words, linear correlation measures the covariance of the corresponding values of series \(X\) and \(Y\) normalized with their standard deviations \(\sigma_X\) and \(\sigma_Y\). It is often more convenient to standardize time series, \(X' = (X - \mathbb{E}(X))/\sigma_X\) and \(Y' = (Y - \mathbb{E}(Y))/\sigma_Y\). And for the standardized series \(X', Y'\) with \(\mathbb{E}(X') = \mathbb{E}(Y') = 0, \sigma_{X'} = \sigma_{Y'} = 1\) the linear correlation formula simplifies to \(\rho(X', Y') = \mathbb{E}(X'Y')\).

Major disadvantage of the correlation coefficient is due to its linearity and inability to capture non-linear relationships in time series. A natural generalization of the linear correlation coefficient that takes into account arbitrary non-linear relationships was first introduced by Gebelein [49] and refined with postulates and proofs by Renyi [50]. Renyi considered a general measure of dependence \(\delta(X, Y)\) and outlined the following postulates:

1) \(\delta(X, Y)\) is defined for any pair of random variables \(X\) and \(Y\). Neither of them being constant with probability 1.
2) It is symmetric, \(\delta(X, Y) = \delta(Y, X)\).
3) \(\delta(X, Y) = 0\) if and only if \(X\) and \(Y\) are independent.
4) \(0 \leq \delta \leq 1\).
5) \(\delta(X, Y) = 1\) only if there is a strict dependence between them.
6) \(\delta(X, Y) = \delta(Id(X), Id(Y))\) where \(Id\) is a Borel-measurable identity function.
7) if the joint distribution of \(X\) and \(Y\) is normal, then the \(\delta(X, Y) = |\rho(X, Y)|\) (\(\rho\) is the linear correlation).

The generalization named maximal correlation coefficient fits the postulates and is defined as

\[
\rho_{\text{max}}(X, Y) = \max_{f,g} \rho(f(X), g(Y)) \tag{28}
\]
where \( f: \mathbb{R} \to \mathbb{R} \) and \( g: \mathbb{R} \to \mathbb{R} \) are Borel-measurable functions, such that \( 0 \leq \text{Var}(f(X)) < \infty \) and \( 0 \leq \text{Var}(g(Y)) < \infty \). Maximal correlation coefficient is minimal and equals to 0 when \( X \) and \( Y \) are independent and is maximal and equals to 1 when there exists a non-linear relationship \( Y = f(X) \) or \( X = g(Y) \), i.e. one of the series is a non-linearly transformed copy of the other.

Figure 8 compares two autocorrelation functions: computed with linear and maximal correlation coefficients computed for 1 month of average hourly CitiBike pick-up time series (720 hours). While the linear correlation coefficient varies in the range \([-0.25, 0.4]\) and captures daily periodicity of the series, the Maximal correlation coefficient detects stronger dependency and complex non-linear relationship taking values in the range \([0.3, 1]\).

The values of maximal autocorrelation exceed corresponding values of the linear autocorrelation for every time lag \( \tau \), \( \rho_{\text{max}}(X_1^{N-\tau}, X_\tau^{N}) > \rho(X_1^{N-\tau}, X_\tau^{N}) \).

Some special cases for the calculation of the maximal correlation coefficient were studied by Dembo et al. \[51\]. They demonstrated that for a collection of independent and identically-distributed random variables \( Y_1, Y_2, \ldots, Y_N \) that are non-degenerate and have a finite variance, \( \text{Var}(Y_i) < \infty \), the maximal correlation between their partial sums \( S_k = \sum_{i=1}^{k} Y_i \) is equal to

\[
\rho_{\text{max}}(S_m, S_n) = \sqrt{\frac{m}{n}} \tag{29}
\]

Another special case was proven by Lancaster \[52\] and later included as Renyi’s 7th postulate for dependency measure \( \delta(X, Y) \) \[50\]: if \( (X, Y) \) is a bivariate Gaussian vector, then

\[
\rho_{\text{max}}(X, Y) = |\rho(X, Y)| \tag{30}
\]

i.e. for the random variables \( X \) and \( Y \) that form a bivariate Gaussian distribution there do not exist Borel-measurable transformations \( f^*(X) \) and \( g^*(Y) \) other than identity functions that maximize the value of the linear correlation.
correlation after $X$ and $Y$ are transformed.

Best of our knowledge a general formula to calculate the maximal correlation coefficient for a pair of continuous random variables does not exist, however, for discrete random variables or time series the formula was derived by Witsenhausen [53]:

$$\rho_{\text{max}}(X, Y) = \lambda_2$$  \hspace{1cm} (31)

where $\lambda_2$ is the second singular value of the normalized joint probability matrix $Q = U\Sigma V^T = (q_{ij})$. The elements of the matrix $q_{ij} = \frac{p_{ij}}{\sqrt{p(i)}\sqrt{p(j)}}$. The probability $p_{ij}$ comes from the contingency table for $X$ and $Y$, and it is equal to $p_{ij} = P(X = \alpha_i | Y = \beta_j)$ with $\alpha_i$ and $\beta_j$ pulled from the ordered set of the unique values that $X$ and $Y$ take, $p(i) = \sum_j p_{ij}$, $p(j) = \sum_i p_{ij}$.

In addition to the calculation of the value of the maximal correlation coefficient, Witsenhausen’s approach provides the numerical definitions for the functions $f^*(X)$ and $g^*(Y)$ that maximize the value of the linear correlation coefficient. The second left and the second right singular vectors contain $f^*(X)$ and $g^*(Y)$ normalized with $\sqrt{p(i)}$ and $\sqrt{p(j)}$ correspondingly.

However, one of the main disadvantages of the maximal correlation coefficient is that by definition the functions $f^*$ and $g^*$ are not guaranteed to have inverse counterparts $(f^*)^{-1}$ and $(g^*)^{-1}$. That means that in general maximal correlation coefficient cannot be used for time series forecasting with autoregressive models because the values of the series will be transformed irreversibly.

Kimeldorf et al. proposed alternative generalization of the linear correlation coefficient [54] - the monotone correlation

$$\rho_{\text{mono}}(X, Y) = \max_{f,g \in \mathcal{F}} (\rho(X, Y))$$  \hspace{1cm} (32)

where $\mathcal{F} = \{ f : \mathbb{R} \rightarrow \mathbb{R} | f \text{ is monotone} \}$, i.e. the supremum is taken over all monotone functions $f$ and $g$ that are Borel-measurable and have finite variances $0 \leq \text{Var}(X) < \infty$ and $0 \leq \text{Var}(Y) < \infty$. Best of our knowledge, there is no general algebraic formula for the monotone correlation coefficient and its calculation is maximization problem. Similarly to the maximal correlation, monotone correlation coefficient takes values in the range $[0, 1]$, and the functions $f^*$ and $g^*$ that maximize the value of the linear correlation coefficient are not guaranteed to have inverses.

Kimeldorf et al. [55] also expanded the notion of monotone correlation with concordant and discordant coefficients and performed their calculation with non-linear optimization. They defined concordant monotone correlation as $\rho_{\text{mono}}(X, Y)$ for which the $f$ and $g$ are both either decreasing or increasing monotone functions. While for the discordant monotone correlation $f$ and $g$ are not both decreasing or increasing at the same time.
The relation between the correlation coefficients is

$$|\rho(X, Y)| \leq \rho_{DMC}(X, Y) \leq \rho_{\text{mono}}(X, Y) \leq \rho_{CMC}(X, Y) \leq \rho_{\text{max}}(X, Y)$$ \hspace{1cm} (33)

where $\rho_{DMC}$ and $\rho_{CMC}$ are the discordant and concordant monotone correlation coefficients correspondingly.

Concluding, we notice that none of the discussed correlation coefficients that rely on maximization over $f$ and $g$ guarantee that they have inverses $(f^*)^{-1}$ and $(g^*)^{-1}$ in general. Therefore, there exists a need for a correlation coefficient that would guarantee that inverse transformations exist.

**B. Clustering**

In this subsection we will consider clustering of a collection of time series. We will denote the collection as $x = \{X_1, X_2, \ldots, X_N\}$ where every time series $X_i = \{X_{i,1}, X_{i,2}, \ldots, X_{i,M}\}$, $X_{i,j} \in \mathbb{R}$. The problem of clustering considers unsupervised partitioning of $x$ into $k$ groups assigning 1 label from the set of labels $\{C_1, C_2, \ldots, C_k\}$ to every time series $X_i$ such that every label $C_i$ is assigned at least once.

Naïve approach to time series clustering when each time series $X_i \in x$ is treated as an unordered vector with clustering algorithms that rely on distance measure (e.g. K-means or K-medoids) might not be efficient if we are interested in grouping time series according to a certain pattern. For example, consider 3 time series on Fig. 9.

![Time series with deterministic linear trend (X₁, X₂) and high stochasticity (X₃)](image)

Figure 9. Time series with deterministic linear trend ($X_1$, $X_2$) and high stochasticity ($X_3$)

Partitioning such time series into two clusters using Euclidean distance to calculate their similarity will result in series $X_2$ and $X_3$ put into one cluster ($C_0$) and $X_1$ into another one ($C_1$). However, what we would really want to achieve is time series $X_1$ and $X_2$ separated from $X_3$ due to their deterministic linear trend.
Such a partition can be achieved if we first calculate linear autocorrelation functions for every time series \( X_i \)

\[
\rho(X_i, L^\tau X_i) \quad 2 \leq \tau \leq N - 1
\]

where \( L^\tau X_i \) stands for the time series \( X_i \) that was shifted with a lag \( \tau \) (\( L^\tau X_i,t = X_i,t-\tau \)). Clustering a collection of autocorrelation functions \( R_{xx} = \{ r_{xx}(X_1), r_{xx}(X_2), r_{xx}(X_3) \} \) will group series \( X_1 \) and \( X_2 \) together because the distances between the corresponding autocorrelation functions:

\[
|| r_{xx}(X_1) - r_{xx}(X_2) || < || r_{xx}(X_1) - r_{xx}(X_3) ||
\]

and \( || r_{xx}(X_1) - r_{xx}(X_2) || < || r_{xx}(X_2) - r_{xx}(X_3) || \).

A similar approach would be to differentiate time series \( X \) prior to clustering, i.e. \( X_i',t = X_i,t - X_i,t-\tau \). Time series with a strong linear trend (\( X_1 \) and \( X_2 \)) would end up having positive values of \( X_i',t \) while the more random series \( X_3 \) will have alternating positive and negative values after differentiation.

In general time series clustering approaches can be divided into 3 classes: raw-data based, feature-based, and model-based. The first class is characterized with a direct application of clustering methods to the raw data. Very often the key modification of the clustering approaches that allows them to partition collections of time series comes in a form of new distance or dissimilarity measures incorporated in standard clustering algorithms.

Komelj and Batagelj [56] proposed a cross-sectional approach that modified relocation clustering method so that it is applicable for time series. The relocation clustering procedure consists of 3 steps:

1) Start with an initial partition \( C \) having \( k \) clusters.

2) Using appropriate dissimilarity measure calculate the dissimilarity matrix.

3) Iteratively look for a better partition \( C' \) by swapping series in cluster \( C_p \) with \( C_q \) until swapping does not improve the current partitioning.

Swapping elements from \( C_p \) to \( C_q \) improves partitioning if and only if \( P(C') < P(C) \) having

\[
P(C) = \frac{1}{2w(C)} \sum_{X,Y \in C} w(X)w(Y)D(X,Y)
\]

\[
w(C) = \sum_{X \in C} w(X)
\]

where \( w(X) \) is a weight of the series \( X \), and \( D(X,Y) \) is the measure of dissimilarity. The proposed modification updated the dissimilarity measure

\[
D(X,Y) = \sum_t \alpha_t d_t(X,Y)
\]

where \( \alpha_t \) satisfies the following conditions to capture temporal patterns: \( \alpha_s \neq \alpha_t \) for \( s \neq t \), \( \alpha_t \geq 0 \), \( \sum_t \alpha_t = 1 \), and \( d_t(X,Y) \) is a dissimilarity measure between two series \( X \) and \( Y \) at time \( t \). Though the dissimilarity measure \( D(X,Y) \) captures temporal relations between individual values of time series \( X \) and \( Y \), the measure does not capture different subsequences \( X_t, X_{t+1}, \ldots, X_{t+k} \) and \( Y_{t-k}, Y_{t-k+1}, \ldots, Y_t \).
Golay et al. [57] applied fuzzy $c$-means clustering algorithm to time series of fMRI. $c$-means clustering algorithm partitions a collection of $N$ time series $x = \{X_1, X_2, \ldots, X_N\}, X_i \in \mathbb{R}^M$ into $c$ fuzzy clusters minimizing the objective function

$$
\arg\min_{C} \sum_{i=1}^{N} \sum_{j=1}^{c} w_{ij}^m ||X_i - c_j||^2
$$

$$
w_{ij} = 1/\sum_{k=1}^{c} \left( \frac{||X_i - c_j||}{||X_i - c_k||} \right)^{2-\frac{1}{m}}
$$

(37)

where $m$ is the hyper-parameter controlling the “fuzziness” of the clusters, $c_i = \sum_{k} w_k(X_i)^m X_i$ are the cluster centroids, and $w_k(X_i)$ quantifies the degree to which $X_i$ belongs to cluster $k$. They proposed two cross-correlation-based distance measures that are applicable to time series

$$
d_{cc}^1(X,Y) = \left( \frac{1 - \rho(X,Y)}{1 + \rho(X,Y)} \right)^{\beta}
$$

$$
d_{cc}^2(X,Y) = 2 \left( 1 - \rho(X,Y) \right)
$$

(38)

where $\rho(X,Y)$ is a linear correlation coefficient, and $\beta$ is a hyper-parameter of the measure. Main disadvantage of the linear correlation based distance measures is in their inability to capture non-linearities in the series.

Yet another modification of the fuzzy $c$-means clustering approach was proposed by Möller-Levet et al. [58]. They introduced short time series (STS) distance that is useful for time series with unevenly spaced sampling points, e.g. DNA microarray data. However, the method is also suitable for series with evenly spaced sampling points. Assuming that time series signal is linear between the measurements, STS compares the difference between the slopes of time series $X$ and $Y$. It is defined as

$$
d_{STS}^2 = \sum_k \left( \frac{\delta X_k}{\delta t_k} - \frac{\delta Y_k}{\delta t_k} \right)^2
$$

(39)

where $\delta X_t = X_t - X_{t-1}$ and $\delta t_k = t_k - t_{k-1}$ are the differences in the observed values of time series and their corresponding time stamps. The limitation of the approach is that it depends on the scale of the time series $X$ and $Y$, i.e. if $\mu_X >> \mu_Y$ and both series have identical linear trend $X_t - X_{t-1} = Y_t - Y_{t-1} + c$ then $d_{STS}(X,Y) \sim c^2$.

Feature based approaches apply some transformations to the original time series and then apply standard clustering algorithms to the transformed series. Vlachos et al. [59] modified standard $k$-means clustering replacing original time series with those transformed with Discrete Wavelet Transform (DWT) using Haar wavelets. They proposed to cluster the decomposed time series at increasingly finer levels of decomposition. The new algorithm, $I-k$-means operates on 5 steps:

1) Decide on a value for the number of clusters $k$. 

20
2) Initialize $k$ cluster centers randomly.

3) Run the $k$-means clustering algorithm on the level $i$ of the representation of the data.

4) Project the final centers that were obtained for level $i$ from space $2^i$ to space $2^{i+1}$ for level $i + 1$.

5) Stop if none of the series were swapped between clusters. Otherwise repeat from step 3.

Fu et al. [5] proposed feature based modification of the self-organizing maps clustering algorithm. The modification of the original time series $X$ was performed with dimensionality reduction through smoothing. The key idea of the method is to keep Perpetually Important Points (PIP) $P$ from the original time series $X_i$ given querying series $Q$ removing the other values. The PIP are selected as follows:

1) Select the first and last points of $P$.

2) Select a point $p_k$ that has the largest distance to its adjacent PIP. Add $p_k$ to the PIP.

3) Repeat step 2 until the minimal distance to adjacent PIP is below the predefined threshold $\lambda$.

The distance to the PIP is measured in terms of Vertical Distance (VD), $VD(p_i, p_j) = |p_i - p_j|$. Once the PIP are selected, the distance between the querying series $Q$ and selected PIP $SP$ is calculated point-wise

$$D(SP, Q) = \mathbb{E}[(T(SP) - T(Q))^2]$$

where $T$ extracts time coordinate from the selected PIP $SP$ and the querying time series $Q$. The major limitation of the approach is its dependence on the predefined pattern set $Q$ which cannot be universally created for an arbitrary collection of time series $x$.

The third class of the time series clustering approaches is the model based class. The main assumption for these approaches is that there is an underlying generating process that can be modeled with certain models (e.g. ARMA and its modifications).

Maharaj [60] proposed a clustering approach based on the $\chi^2$ test statistic that given two time series $X$ and $Y$ assuming that they are generated by an autoregressive process of order $k$, AR($k$) with parameters $\theta^X = \{\theta_1^X, \ldots, \theta_k^X\}$ and $\theta^Y = \{\theta_1^Y, \ldots, \theta_k^Y\}$, and setting the null hypothesis $H_0 : \theta^X = \theta^Y$, cluster $X$ and $Y$ if the $p$-value is greater than the predefined threshold. The main weakness of the approach is due to the simplicity and linearity of the AR($k$) model.

C. Summary

To overcome the limitations of the linear correlation coefficient, it was generalized with maximal, monotone, concordant, and discordant correlation coefficients [49] [50] [54] [55]. Though the generalized coefficients are very useful in measuring non-linear dependency, they are not applicable for autoregression models because the transformation used for their calculations are not guaranteed to have inverses. Monotone correlation coefficient and its variants are computed as a maximization problem which might be costly for very long time series.
Clustering methods can be classified into 3 major groups: methods that cluster raw data, methods that generate features, and methods that fit models. Methods in the first group modify similarity or distance measures to make them suitable for time series and then apply standard clustering methods [56] [57] [58]. However, the majority of the proposed distance measures compare the values of time series pair-wise and don’t match subsequences. Feature-based methods generate new features transforming the series [59] [5] (e.g. Fourier, Wavelet), computing autocorrelation functions, reducing dimentionality, etc., and then clustering the results. However, such methods require prior feature engineering. Model-based clustering approaches [60] fit a certain model (often linear autoregression) to time series and then cluster them based on hypothesis testing or parameters of the model. Major limitations are due to the selected model: the better the model fits the series, the more efficient is clustering.

Overall, the potential research directions include: study of the new non-linear correlation coefficients that use inverse transformations, exploration of fast approximations of monotone correlation coefficient, use of non-linear correlation coefficients to measure similarity between two time series.

IV. TIME SERIES FORECASTING

Time series forecasting has long been of interest to the researchers. Over time numerous statistical techniques were developed from fitting simple linear models, decomposition and extrapolation with Fourier or Wavelet transforms, to training complex neural networks like Long Short-Term Memory (LSTM) and other complex machine learning techniques.

A. Lempel-Ziv-Welch

The LZW predictor for time series is based on the Lempel-Ziv-Welch text encoding algorithm [32]. Given a time series \( X = \{X_1, X_2, \ldots, X_N\} \) LZW algorithm partitions it into a collection of distinct subsequences \( S_0, S_1, \ldots, S_M \) where \( S_k \) represents the shortest subsequence that starts at the time \( k \) which has not been seen at prior time intervals in \( X \). For example, given time series \( X = \{1, 1, 2, 1, 1, 2, 1, 3, 2\} \) LZW algorithm partitions it into a set of subsequences

\[
\{\{1\}, \{2\}, \{3\}, \{1, 1\}, \{1, 2\}, \{2, 1\}, \{1, 2, 1\}, \{1, 3\}, \{3, 2\}\}
\]

(41)

Figure [10] shows the resulting LZW-tree of the extracted shortest subsequences \( S_k \) seen at time \( k \).

LZW-tree grows dynamically during the parsing of time series \( X \). The root of the tree is an empty set. Starting from the root each node represents one subsequence \( S_k \) of the original series \( X \) with the sequences of nodes encountered on the path to the node. Each node also stores a counter of the occurrences of the subsequences.
Overall, the LZW-prediction of the future value of time series $X$ at time $t + 1$ given the historical values $X_t = \{X_1, X_2, \ldots, X_t\}$ is defined as

$$P(X_{t+1} = \beta | X_t) = \frac{c(\{S_{1,1}, S_{1,2}, \ldots, S_{k,M}, \beta\} | X_t)}{c(S_k | X_t)}$$

(42)

where $c$ stands for the count, i.e. the probability of observing a value $\beta$ at time period $t + 1$ is the ratio of the number of times the shortest subsequence $S_k$ was followed by $\beta$ normalized with the number of times $S_k$ occurred in time series $X$. The key disadvantage and limitation of the approach is that it is combinatorial in essence and relies on the presence of repetitions. The values are treated as symbols and the magnitude of the values is not taken into account. Another problem with LZW-predictor could arise in the presence of noise in time series, $X = Y + Z$ where $Z \sim \mathcal{N}(\mu, \sigma^2)$. For example, given time series $Y = \{0, 0, 1, 0, 2, 0, 0, 1\}$ and noise series $Z = \{0, 0.2, 0.1, 0.1, 0.15, 0.2, 0.1, 0, 0.2\}$, LZW-predictor will have a lower forecasting accuracy for the noisy time series $X = \{0, 0.2, 1.1, 0.1, 0.15, 0.2, 0.1, 0, 1.2\}$ compared to time series $Y$. As a consequence, it might be necessary to round the values $X_t$ of time series $X$ prior to forecasting it with LZW algorithm.

A modification of LZW-predictor was proposed by Feder et al. [61] in their study that addressed binary sequence forecasting. They introduced a finite-state predictor based on the incremental parsing procedure of the Lempel-Ziv compression algorithm. They defined a finite-state predictability measure and demonstrated that the proposed algorithm asymptotically approaches it.

Often LZW-predictor and its variations appear in research papers related to transportation domain, e.g. forecasting taxi demand or human mobility.

**B. Markov chains**

Markov chain predictor is based on a similar idea - estimation of the frequencies of observed subsequences. Given time series $X = \{X_1, X_2, \ldots, X_N\}$ the order-$k$ ($O(k)$) Markov chain predictor [62] [63] makes a prediction
The predictor relies on the Markov assumption

\[ P(X_{t+1} = \beta | X_1^t) = P(X_{t+1} = \beta | X_{t-k+1}^t) = P(X_{t+k+1} = \beta | X_{t+1}^t) \]  

(43)

where \( P(X_{t+1} = \beta | X_1^t) \) stands for the probability of observing the value \( \beta \) in the future at time interval \( t + 1 \) having the historical series \( X_1^t = \{X_1, X_2, \ldots, X_t\} \). The key idea of the predictor is that equal subsequences \( X_{t-k+1}^t = \{X_{t-k+1}, X_{n-k+2}, \ldots X_t\} = X_{t+1}^{t+k} = \{X_{t+1}, X_{t+2} \ldots X_{t+k}\} \) are likely to be followed by the same value \( X_{t+1} = X_{t+k+1} = \beta \).

Denoting the repeating subsequences \( X_{t-k+1}^t = X_{t+1}^{t+k} = S = \{S_1, S_2, \ldots, S_k\} \), the estimate probability that the future value of the time series \( X \) at time period \( t + 1 \) is equal to \( \beta \), \( P'(X_{t+1} = \beta | X_1^t) \) given the observed values of \( \{X_1, \ldots, X_t\} \) is calculated as

\[ P'(X_{t+1} = \beta | X_1^t) = \frac{c(\{S_1, S_2, \ldots, S_k, \beta\}, X_1^t)}{c(S, X_1^t)} \]  

(44)

where \( c(\{S_1, \ldots, S_k, \beta, X_t\}) \) is the number of occurrences of the subsequence \( S \) followed by \( \beta \), and \( c(S, X_t) \) represents the total count of appearances of \( S \) in time series \( X_1^t \). Their ratio corresponds to the probability of observing \( S \) followed by \( \beta \) in the future.

Markov chain predictor and its derivatives have a limitation that is similar to LZW predictors: they rely on the presence of repetitions and severely underperform in the presence of noise in time series. While the advantage of both Markov chains and LZW is the computational simplicity and high efficiency for time series that are highly periodic and contain numerous repeating subsequences.

Markov chain models and their modifications are extensively used for short-term traffic flow forecasting [64], synthetic wind speed time series generation with first and second order models [65]. Another application of Markov chains is approximate forecasting that is later improved with more complex neural network models [66].

C. Classifiers and Support Vector Machines

A different approach to time series forecasting is to turn time series forecasting into a classification or regression problem, i.e. given feature vectors \( x \in \mathbb{R}^{m \times n} \) and corresponding labels \( y \in \mathbb{R}^n \), find parameters \( \alpha \) of a certain model \( f \), such that \( f(X_i, \alpha) = y_i \) for all \( i \). In this subsection we will consider Software Vector Machines as our classifier, however, the approach is applicable to any similar classifier or regressor in general.

The key idea is as follows: we partition time series \( X \) into \( k \) overlapping subseries of size \( p \) creating feature vectors \( x \in \mathbb{R}^{(p-1) \times (N-p+1)} \) and label vector \( y \in \mathbb{R}^{N-p+1} \)

\[ x = (X_1^{p-1}, X_2^p, \ldots, X_{N-p+1}^{N-1}) \]

\[ y = X_p^N \]  

(45)
where $p$ is periodicity of the series $X$ (the size of the time lag) that is determined with either autocorrelation function or Fourier transformation. And then we train a classifier model in a regular way passing $x$ as features and $y$ as the label.

For example, we can use the Support Vector Machines (SVM) as classifier (originally it was proposed by Vapnik [67]). Most of the modern SVM-based predictors fit the soft-margin model [68] that for our features $x$ and label $y$ minimizes

$$\frac{1}{N - p + 1} \sum_{i=1}^{N-p+1} \max(0, 1 - y_i(w^TX_i - b)) + \lambda||w||^2$$

(46)

where $w^TX_i - b$ determines the separating hyperplane, and $\lambda$ regulates the trade-off between the size of the margin and making sure that points $X_i$ lie on the proper side of the separating hyperplane, i.e. they are classified correctly.

The initial SVM and soft margin SVM models assumed that the points in $x$ are linearly separable which is not true in general. Boser et al. [69] generalized the model with addition of non-linear kernels that effectively transform the feature vectors $x$ allowing to separate low dimensional non-linearity with a linear hyperplane in higher dimensions. Most frequently used kernels are polynomial of degree $d$, $K(X_i, X_j) = (X_iX_j + 1)^d$, and radial (Gaussian) $K(X_i, X_j) = \exp(-1/\delta^2(X_i - X_j)^2)$.

A successful relatively recent application of the SVM model to engine reliability predictions was performed by Hong et al. [70]. More recently an improved SVM model was applied to financial time series forecasting by Tay and Cao [71].

Among the major disadvantages of time series forecasting with classifiers in general (including SVM) are the inability of capturing temporal patterns within the subsequences $X_i$ that are treated as unordered vectors, and the inability to forecast a value $\hat{X}_{t+1} = \beta$ that was not observed historically in the series $X_t$. For example, if the series $X$ has a significant positive trend, then the forecasts created with classifiers will decrease in accuracy making systematic underpredictions.

D. ARMA and its generalizations

Historically, researchers applied simple linear models for time series forecasting - autoregression and moving-average. An autoregressive model of order $p$ is defined as

$$X_t = c + \sum_{i=1}^{p} \phi_i X_{t-i} + \epsilon_t$$

(47)

where $\phi = \{\phi_1, \ldots, \phi_p\}$ and $c$ are the model’s parameters, and $\epsilon_t$ is an i.i.d. error at time $t$ often distributed normally, $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$. Moving-average model is essentially a linear regression of the values of time series $X$
at time $t$ against $q$ error terms

$$X_t = \mu + \epsilon_t + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} \quad (48)$$

where $\mu$ is the mean of $X$, $\theta = \{\theta_1, \ldots, \theta_q\}$ are the model’s parameters, and $\epsilon = \{\epsilon_1, \epsilon_2, \ldots, \epsilon_N\}$ are the noise parameters. Similarly to the autoregressive noise variables, $\epsilon_i \sim N(0, \sigma^2)$.

A combination of the models first proposed by P. Whittle [72] later being extensively studied by Box and Jenkins [73] was named ARMA (autoregressive moving-average)

$$X_t = c + \epsilon_t + \sum_{i=1}^{p} \phi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} \quad (49)$$

It is often more convenient to rewrite the model with the use of lag operator $L$ of order $k$ [73]

$$L^k X_t = X_{t-k} \quad (50)$$

Replacing $X_{t-i}$ with $L^i X_t$ and reorganizing the expression for the ARMA model we have

$$\left(1 - \sum_{i=1}^{p} \phi_i L^i\right) X_t = \left(1 + \sum_{i=1}^{q} \theta_i L^i\right) \epsilon_t \quad (51)$$

The most widely used generalization of ARMA model is ARIMA (autoregressive integrated moving-average) that introduces nonstationarity to the model

$$\left(1 - \sum_{i=1}^{p} \phi_i L^i\right) (1 - L)^d X_t = \delta + \left(1 + \sum_{i=1}^{q} \theta_i L^i\right) \epsilon_t \quad (52)$$

where $d \in \mathbb{N}$ stands for the multiplicity of a unit root $(1 - L)$, $\delta$ introduces the drift of the model that is equal to $\delta / \sum_i \phi_i$.

In general the order of the model is identified as ARIMA$(p, d, q)$ where $p$ is the number of auto-regressive terms, $d$ is the number of nonseasonal differences (integration) turning the model into a nonstationary one with $d \neq 0$, and $q$ is the number of error terms (moving-average).

A standard way to identify the proper order of the model is to the Akaike Information Criterion (AIC) [74]:

$$AIC(p, d, q) = -2 \log(\mathcal{L}) + 2(p + q + k + 1) \quad (53)$$

where $\mathcal{L}$ is the maximum likelihood estimator of the innovation variance, $k \in \{0, 1\}$ stands for the presence or absence of the constant term in the model, and the sum $p + q + k + 1$ represents the total number of parameters in the model.

The optimal model ARIMA$(p, d, q)$ is selected as the one that has the lowest $AIC$ coefficient compared with the other orders considered. To improve the prediction accuracy and decrease the number of overall iterations, one might consider relating the choice of the parameters $p$ and $q$ with the inverse frequencies of largest magnitudes.
in the frequency spectrum of the original time series \( X \). Alternative approaches to select the parameters \((p, d, q)\) minimize \( \sum_{i=1}^{N} \epsilon_i^2 \).

Often a generalization of ARIMA model called Fractional ARIMA that allows for the integration parameter \( d \) to take continuous values in range \((-0.5, 0.5)\) is used for traffic forecasting \[75\].

Another generalization, Seasonal ARIMA makes an emphasis on the seasonality of the series \( X \). SARIMA model of order \((p, d, q) \times (P, D, Q)\) is defined as \[73\]

\[
(1 - \sum_{i=1}^{p} \phi_i L^i)(1 - \sum_{i=1}^{P} \Phi_i L^is) (1 - L)^d (1 - L^s)^D X_t = (1 + \sum_{i=1}^{q} \theta_i L^i)(1 + \sum_{i=1}^{Q} \Theta_i L^is) \epsilon_t
\]

where \( s \) is the seasonal lag, \( \Theta = \{\Theta_1, \ldots, \Theta_q\} \), \( \Phi = \{\Phi_1, \ldots, \Phi_p\} \), and \( D \) are the seasonal counterparts of the original \((p, d, q)\) parameters of ARIMA model.

The key disadvantage of the ARIMA family of the models is their inability to capture non-linear patterns in time series. For example, Moreira-Matias et. al. built a framework for taxi-passenger demand forecasting using ARIMA, Weighted and Unweighted time varying poisson, and their ensemble \[76\]. ARIMA was demonstrated to have the worst forecasting accuracy among the implemented models.

In order to apply ARIMA models to time series with non-linear components, some researchers combine it with SVM (Chen and Wang \[77\]), while the others integrate it with neural networks (Zhang \[78\]).

**E. Neural Networks**

Over the past few decades neural network (NN) models have gained a lot of attention. Being universal approximators \[79\], neural networks are able to capture complex non-linear patterns and are applied for classification, regression, compression, and numerous other applications. Following our definition for the construction of feature vectors \( x \) and label vector \( y \) from the values of time series \( X \), we have \( x = (X_{p-1}^1, X_2^p, \ldots, X_{N-p+1}^{N-1}) \), \( y = X_p^N \).

Then the simplest neural network model learns the function

\[
f(X_i) = w_2^T \phi(w_1^T X_i + b_1) + b_2
\]

where \( w_2 \in \mathbb{R}^m \) is the hidden layer weights of \( m \) neurons, \( w_1 \in \mathbb{R}^{p-1} \) stands for the input layer weights of \( p - 1 \) neurons, \( \phi : \mathbb{R} \rightarrow \mathbb{R} \) is the activation function, and \( b_2, b_1 \) represent the bias of the hidden and the input layers correspondingly. Most widely used activation functions include hyperbolic tangent \( \tanh \), ReLU, softmax \( \sigma \), and sigmoid \( \sigma \).

One of the major advantages of neural networks is their ability to increase the complexity of the model simply increasing the number of hidden layers and neurons per layer. On the other hand, selecting the optimal topology
of the network and picking the right number of parameters is a research problem of its own. Lam et al. [80] proposed modified genetic algorithm that learns the optimal number of layers and neurons per layer.

Another common problem with neural networks is overfitting of the model. Some of the solutions include removal of excessive neurons [81]. Other solutions include regularization [82]

\[ H[f] = (f(X_i) - y_i)^T (f(X_i) - y_i) + \lambda \phi[f] \]  

where \( f \) is the function we are aiming to learn, \( \phi \) is the smoothness functional, and \( \lambda > 0 \) is the regularization parameter.

Figure 11 demonstrates an example of a neural network topology that was used for taxi-demand forecasting by Zhao et al. [4]. The network contains 2 hidden layers with Sigmoid and Softmax activation functions.

![Figure 11. Topology of a simple neural network predictor with 2 hidden layers](image)

In the work taxi-demand time series \( X \) were combined with exogenous time series \( T \) that captured the weather data including precipitation, temperature, wind speed, and calendar data including day of the week, and hour of the day. The outputs of the neural network were additionally smoothed to further prevent it from overfitting. Linear convolution with the Gaussian kernel was applied:

\[ G(\hat{X}_{t+1}) = \sum_i \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(\hat{X}_{t+1} - x_i)^2}{2\sigma^2}} \]  

where \( \hat{X}_{t+1} \) is the value of time series \( X \) at time \( t + 1 \) forecasted by the neural network model. Combining the output of a neural network with linear convolution of the previous values of the time series can improve the predictor’s ability to capture temporal patterns within the feature vectors \( x \).

A common deep neural network model that is generally used for denoising and compression of images was found to be applicable and efficient for time series forecasting as well [83]. The idea behind autoencoders is to reproduce the input through encoding it into a hidden representation and then decoding it back to original form.
Figure 12 shows a sample schema of an autoencoder that compresses 4 input neurons into a hidden representation with 3 neurons.

The simplest autoencoder model that only has two layers - encoder and decoder can be defined as

\[
\begin{align*}
y(x) &= f(w_1^T x + b_1) \\
z(x) &= g(w_2^T y(x) + b_2)
\end{align*}
\]

where \(y(x)\) encodes the original features \(x\) into the hidden representation, and \(z(x)\) decodes \(y(x)\) back to original features \(x\). \(b_1\) and \(b_2\) stand for encoding and decoding bias vectors, \(f\) and \(g\) are the encoding the decoding activation functions (often sigmoid or softmax).

Model parameters \(\theta\) are obtained through minimization of the reconstruction errors \(L(x, z(x))\)

\[
\theta = \underset{\theta}{\text{arg min}} L(x, z(x)) = \underset{\theta}{\text{arg min}} \frac{1}{2} \sum_{i=1}^{N} ||x_i - z(x_i)||^2
\]

Using this principle deep neural network is constructed through feeding the output from an autoencoder of one layer into the input of another. The final output layer usually contains a predictor, e.g. logistic regression. However, even classic neural networks and deep autoencoders do not take the temporal order of the features \(x\) into account.

A different type of neural networks emerged from non-linear generalization of autoregressive models AR(\(P\)) discussed in ARMA subsection. Non-linear autoregression model or NAR

\[
X_t = h(X_{t-1}, X_{t-2}, \ldots, X_{t-p}) + \epsilon_t
\]

where \(h\) is a smooth function, \(\epsilon_t\) is a noise with \(\mu_\epsilon = 0\) and \(\sigma_\epsilon^2 < \infty\). Often it is assumed that \(\epsilon_t \sim N(0, \sigma^2)\).

Lapedes and Farber proposed a feed forward neural network model \([84]\) that approximated the NAR model’s non-linear functions \(h\)

\[
X_t = h(X_{t-1}, \ldots, X_{t-p}) = \sum_{i=1}^{I} W_i f \left( \sum_{j=1}^{p} w_{ij} X_{t-j} + \theta_i \right)
\]
where $W_i$ and $w_{ij}$ are the weights for the connections to the neurons of the output and hidden layer correspondingly. The neural network approximation of NAR models was the first step towards creating a recurrent neural network that employs the temporal relationship among the consecutive values of $X$.

Connor and Martin [85] demonstrated that the non-linear generalization of ARMA model, NARMA $(p, q)$ approximated with a neural network is a recurrent neural network (RNN) that can itself be generalized even further into a fully interconnected RNN

$$X_t = \sum_{i=1}^{I} W_i g_i(t)$$  \hspace{1cm} (62)

where the $I$ denotes the total number of hidden inputs and $g_i(t)$ are computed recursively as

$$g_i(t) = f \left( \sum_{j=1}^{\max(p, q)} \tilde{w}_{ij} X_{t-j} + \sum_{k=1}^{q} \sum_{l=1}^{I} \tilde{w}_{ilk} g_l(t-k) + \theta_i \right)$$  \hspace{1cm} (63)

Network topology of the fully interconnected RNN is shown on Figure [13].

![Figure 13. Fully interconnected RNN. Source [85]](image)

Given all the advantages of the proposed RNN models in their ability to capture temporal patterns, their main disadvantage, however, is their susceptibility to the presence of outliers in time series $X$. Such models require outlier filtering prior to their training.

Yet another approach to constructing an RNN through turning individual neurons into complex structures that have "memory" of their previous states was proposed by Hochreiter and Schmidhuber [86]. Analyzing previous work including Time Delay Neural Networks (TDNN), Kalman Filters, NARX, and numerous others, they concluded that the existing models are impractical for storage of long-term minimal time lags.
The proposed new model named Long Short-Term Memory (LSTM) introduced memory cells and gate units. The key idea of the LSTM model is to use neurons that are represented by blocks that have internal structure: input gate, output gate and forget gate which allows them to keep the previous inputs in memory in the order of their appearance, and yet protect the current memory content from previous or current irrelevant perturbations, for example, many RNN models are subject to vanishing or exploding gradients for inputs that are separated by large time lags $\tau$ ($X_t$ and $X_{t+\tau}$).

Figure 14. Internal structure of LSTM cells.

Figure 14 shows the internal structure of LSTM cells and the general topology of LSTM networks. Having the time series $X = \{X_1, \ldots, X_N\}$ we show 3 consecutive values $X_{t-1}$, $X_t$, and $X_{t+1}$ that correspond to 3 cells enumerated as $t−1$, $t$, and $t+1$. $h_t$ is an output of the $t^{th}$ cell, $\otimes$, $\oplus$, and $\tanh$ inside an ellipse represent pointwise product, sum, and $\tanh$ function correspondingly. $\sigma$ and $\tanh$ inside rectangles denote standard neural network layers with sigmoid and hyperbolic tangent activation.

The inputs $X_t$ enter LSTM cells that contain 3 gates within their structures: input, output and forget. In the input gate the original value $X_t$ is first passed through the layers with $\sigma$ and $\tanh$ activation:

$$
\begin{align*}
    i_1 &= \tanh (b^i_1 + X_t W_{i1}^i + h_{t-1} W_{i2}^i), \\
    i_2 &= \sigma (b^i_2 + X_t W_{i2}^i + h_{t-1} W_{i2}^i) \\
\end{align*}
$$

(64)

where $b^i_1$ and $b^i_2$ are the input biases, $W_{i1}^i$ and $W_{i2}^i$ stand for the input weights, $W_{i1}^2$ and $W_{i2}^2$ are the weights of the previous cell’s output $h_{t-1}$. The output of the input gate is the result of pointwise multiplication of the outputs of the $\tanh$ and sigmoid layers, $i_1 \otimes i_2$. 

31
The forget gate is evaluated by another neural network layer with sigmoid activation

\[ f = \sigma(b^f + X_tW^f_1 + h_{t-1}W^f_2) \] (65)

where \( b^f, W^f_1 \), and \( W^f_2 \) represent the forget gate bias, and the forget gate weights for the current cell \( t \) and the previous cell \( t-1 \) correspondingly. The combined output of the input and forget gates is equal to \( f + i_1 \otimes i_2 \). The output gate is yet another layer with \( \sigma \) activation

\[ o = \sigma(b^o + X_tW^o_1 + h_{t-1}W^o_2) \] (66)

Finally, the output of the entire LSTM cell \( t \) is \( h_t = \tanh(f + i_1 \otimes i_2) \otimes o \).

Song et al. [87] built a deep LSTM learning architecture suitable for forecasting of long sequences and able to determine the optimal time lags automatically.

Similarly to most of the other deep and complex NN models, LSTM networks require significantly more training time comparing to simpler models, and might not be suitable for highly predictable time series.

**F. Summary**

Lempel-Ziv-Welch and Markov chains are among the most well known and natural sequential forecasting methods. Both of them calculate relative frequencies of repeating subsequencies within time series. Their major limitation is the decrease of forecasting accuracy in the presence of noise. Both predictors are often used in combination with SVM or Neural Networks to improve their performance.

Machine learning approach to time series forecasting based on overlapping feature and label generation and training a model like Support Vector Machines or a Neural Network is not able to capture temporal patterns within the features and will return identical models if the features are randomly scrambled.

Major limitation for ARIMA family of models is due to their linearity. In some cases ARIMA models can be unstable. Often they are combined with Neural Networks or SVM models with non-linear kernels to capture non-linearity in the time series.

Recurrent Neural Networks are among the most advanced models that are able to fully recognize temporal order within time series. Major limitations of fully connected RNN is caused by them being impractical for storage of long-term minimal time lags. Long Short-Term Memory networks are designed to address the problem having the neurons with internal structure and ability to remove irrelevant perturbations and keep only the important values of time series.

The future research could include incorporation of non-linear correlation coefficients into ARIMA models instead or running optimization for non-linear autoregressive models (NAR) which could improve the computation
time. Another direction could be the study of connection between Neural Networks and the maximal correlation coefficient.

V. CONCLUSION. FUTURE STEPS

In the survey we have analyzed most widely used measures of time series complexity, entropy, predictability, periodicity, similarity, clustering techniques, and forecasting methods. We have identified the following limitations:

- Combinatorial estimators of complexity and entropy including Lempel-Ziv and permutation entropy treat time series like a sequence of symbols and do not take actual values into account and are computationally expensive and impractical for very long series. Generalizations of permutation entropy mitigate the first limitation.
- Fourier and Wavelet based methods for periodicity estimation are not linear in period. Linear autocorrelation function does capture non-linear periodic patterns in time series.
- Non-linear correlation coefficients use transformations that are not guaranteed to have inverses which makes them inapplicable for non-linear autoregression models and time series forecasting. Monotone correlation is computed as a maximization problem.
- Majority of the distance measures proposed for time series clustering compare values pairwise and do not take order within subsequencies of time series into account.
- Machine learning approach to clustering and time series forecasting does not take order within features into account.

To address the discovered limitations, the following future steps should be considered:

- Study of non-linear autocorrelation functions that would generalize linear autocorrelation functions and will be linear in period.
- study of the new non-linear correlation coefficients that use inversible transformations, exploration of fast approximations of monotone correlation coefficient.
- Develop a time series similarity measure based on non-linear correlation coefficients to improve time series clustering techniques.

REFERENCES


